Learning Intrusion Detection: Supervised or Unsupervised?

Pavel Laskov, Patrick Düssel, Christin Schäfer, and Konrad Rieck

Fraunhofer-FIRST.IDA, Kekuléstr. 7, 12489 Berlin, Germany
{laskov, duessel, christin, rieck}@first.fhg.de

Abstract. Application and development of specialized machine learning techniques is gaining increasing attention in the intrusion detection community. A variety of learning techniques proposed for different intrusion detection problems can be roughly classified into two broad categories: supervised (classification) and unsupervised (anomaly detection and clustering). In this contribution we develop an experimental framework for comparative analysis of both kinds of learning techniques. In our framework we cast unsupervised techniques into a special case of classification, for which training and model selection can be performed by means of ROC analysis. We then investigate both kinds of learning techniques with respect to their detection accuracy and ability to detect unknown attacks.

1 Introduction

Intrusion detection techniques are usually classified into misuse detection and anomaly detection [1]. Anomaly detection focuses on detecting unusual activity patterns in the observed data [2,3,4,5,6]. Misuse detection methods are intended to recognize known attack patterns. Signature-based misuse detection techniques are currently most widely used in practice; however, interest is growing in the intrusion detection community to application of advances machine learning techniques [7,8,9,10]. Not uncommon is also a combination of anomaly and misuse detection in a single intrusion detection system.

To decide which learning technique(s) is to be applied for a particular intrusion detection system, it is important to understand the role the label information plays in such applications. The following observations should be considered:

1. Labels can be extremely difficult or impossible to obtain. Analysis of network traffic or audit logs is very time-consuming and usually only a small portion of the available data can be labeled. Furthermore, in certain cases, for example at a packet level, it may be impossible to unambiguously assign a label to a data instance.
2. In a real application, one can never be sure that a set of available labeled examples covers all possible attacks. If a new attack appears, examples of it may not have been seen in training data.

F. Roli and S. Vitulano (Eds.): ICIAP 2005, LNCS 3617, pp. 50–57, 2005.

The main goal of this work is to investigate the tradeoffs between supervised and unsupervised techniques in their application to intrusion detection systems. To this end, we develop an experimental setup in which such techniques can be fairly compared. Our setup is based on the well-known KDD Cup 1999 data set [11]. Although this data set is known to have certain drawbacks, caused by the artificial nature of underlying data in DARPA IDS evaluations, no other data sets are currently available for comprehensive experimental studies. Since a typical application of a supervised learning method involves model selection, we have built in model selection into unsupervised methods. Performance of both groups of methods is evaluated based on the analysis of the receiver operator characteristic (ROC) curve. The details of our experimental setup are presented in Sec. 2.

Evaluation of several representative supervised and unsupervised learning algorithms, briefly reviewed in Sec. 3, is carried out under the following two scenarios. Under the first scenario, an assumption that training and test data come from the same (unknown) distribution is fulfilled. Under the second scenario, we violate this assumption by taking a data set in which attacks unseen in training data are present in test data. This is a typical scheme to test the ability of an IDS to cope with unknown attacks. The experimental results are presented in Sec. 4.

2 Experimental Setup

2.1 Data Source

The KDD Cup 1999 data set [11] is a common benchmark for evaluation of intrusion detection techniques. It comprises a fixed set of connection-based features. The majority of instances in this set (94%, 4898430 instances) has been extracted from the DARPA 1998 IDS evaluation [12]. The remaining fraction of data (6%, 311029 instances) was additionally extracted from the extended DARPA 1999 IDS evaluation [13]. A detailed description of the available features and attack instances can be found in [14,6].

2.2 Preprocessing

The KDD Cup data set suffers from two major flaws in distribution of data which can bias comparative experiments:

1. The attack rate within the KDD Cup data set is unnatural. About 80% of all instances correspond to attacks, since all one-packet attacks, e.g. the `smurf` attack, are treated as full-value connections and are represented as individual instances.
2. The attack distribution within the KDD Cup data set is highly unbalanced. It is dominated by probes and denial-of-service attacks, which cover millions of instances. The most interesting and dangerous attacks, e.g. the `phf` or `imap` attacks, are grossly under-represented.

Table 1. Distribution of attack types in the experiments

"Known" Attack Types	"Unknown" Attack Types
back buffer_overflow ftp_write guess_passwd imap ipsweep land loadmodule multihop neptune nmap perl phf pod portsweep rootkit satan smurf spy teardrop warezclient warezmaster	apache2 httptunnel mailbomb mscan named processtable ps saint sendmail snmpgetattack snmpguess sqlattack updstorm worm xlock xsnoop xterm

In order to cope with these artifacts we preprocess KDD Cup data in order to achieve (a) a fixed attack rate and (b) a balanced distribution of attack and service types.

At the first level of preprocessing, the attack data is split into disjoint partitions containing only one attack type. The normal data is split into disjoint partitions containing only one service type. These partitions are merged into the three disjoint sets of equal length: the training data D_{train}, the validation data D_{val} and the test data D_{test}. This procedure ensures the presence of each attack and service type in the three data partitions.

At the second level of preprocessing samples of 2000 instances are randomly drawn from the training, validation and testing data sets. The sampling procedure enforces a fixed attack rate of 5% and attempts to preserve balanced attack and service type distributions.

The data with "known attacks" is generated from the DARPA 1998 part of the KDD Cup data set. The data with "unknown attacks" has the test part sampled from the DARPA 1999 part of the KDD Cup data set. The attacks in both data sets are listed in Table 1.

2.3 Metric Embedding

The set of features present in the KDD Cup data set contains categorical and numerical features of different sources and scales. An essential step for handling such data is *metric embedding* which transforms the data into a metric space. Our embedding is a two-stage procedure similar to [3,2].

Embedding of Categorical Features. Each categorical feature expressing m possible categorical values is transformed to a value in $\mathbb{R}^m$ using a function e that maps the j-th value of the feature to the j-th component of an m-dimensional vector:

$$e(x_i) = \underbrace{(0, \ldots, 1, \ldots, 0)}_{1 \text{ at Position } j} \quad \text{if } x_i \text{ equals value } j$$

Scaling of Features. Both the numerical and the embedded categorical features are scaled with respect to each feature's mean μ and standard deviation σ:

$$n(x_i) = \frac{x_i - \mu}{\sigma}$$

2.4 Model Selection

Model selection is performed by training a supervised algorithm on a training set D_{train} and evaluating the accuracy on 10 validation sets D_{val} generated as described in Sec. 2. For unsupervised algorithms only evaluation is performed. The criterion for evaluating the accuracy is the area under the ROC curve, computed for the false-positive interval [0, 0.1].

3 Methods

In the following we briefly describe the algorithms used in our experiments.

3.1 Supervised Algorithms

C4.5. The C4.5 algorithm [15] performs inference of decision trees using a set of conditions over the attributes. Classification of new examples is carried out by applying the inferred rules. Although the original algorithms contains numerous free parameters, only the number of bootstrap iterations was used in our evaluation.

***k*-Nearest Neighbor.** The k-Nearest Neighbor is a classical algorithm (e.g. [16]) that finds k examples in training data that are closest to the test example and assigns the most frequent label among these examples to the new example. The only free parameter is the size k of the neighborhood.

Multi-layer Perceptron. Training of a multi-layer perceptron involves optimizing the weights for the activation function of neurons organized in a network architecture. The global objective function is minimized using the RPROP algorithm (e.g. [17]). The free parameter is the number of hidden neurons.

Regularized Discriminant Analysis. Assuming both classes of examples are normally distributed, a Bayes-optimal separating surface is a hyperplane (LDA), if covariance matrices are the same, or a quadratic surface otherwise (QDA). A gradual morph between the two cases can be implemented by using a regularization parameter γ [18]. Another free parameter λ controls the addition of identity matrix to covariance matrices.

Fisher Linear Discriminant. Fisher Linear Discriminant constructs a separating hyperplane using a direction that maximizes inter-class variance and minimized the intra-class variance for the projection of the training points on this direction (e.g. [16]). The free parameter is the tradeoff between the norm of the direction and the "strictness" of projection.

Linear Programming Machine and Support Vector Machine. Linear Programming Machine (LPM) and Support Vector Machine (SVM) construct a hyperplane of the minimal norm which separates the two classes of training examples (e.g. [19]). LPM uses the 1-norm, SVM uses the 2-norm. Furthermore,

SVM apply a non-linear mapping to construct a hyperplane in a feature space. In our experiments, radial basis functions are used, their complexity controlled by the width parameter w. Another parameter C controls the tradeoff between the norm of a hyperplane and the separation accuracy.

3.2 Unsupervised Algorithms

γ-Algorithm. The γ-algorithm is a recently proposed graph-based outlier detection algorithm [20]. It assigns to every example the γ-score which is the mean distance to the example's k nearest neighbors. The free parameter is k.

***k*-Means Clustering.** k-Means clustering is a classical clustering algorithm (e.g. [16]). After an initial random assignment of example to k clusters, the centers of clusters are computed and the examples are assigned to the clusters with the closest centers. The process is repeated until the cluster centers do not significantly change. Once the cluster assignment is fixed, the mean distance of an example to cluster centers is used as the score. The free parameter is k.

Single Linkage Clustering. Single linkage clustering [2] is similar to k-Means clustering except that the number of clusters is controlled by the distance parameter W: if the distance from an example to the nearest cluster center exceeds W a new cluster is set.

Quarter-Sphere Support Vector Machine. The quarter-sphere SVM [5,6] is an anomaly detection method based on the idea of fitting a sphere onto the center of mass of data. An anomaly score is defined by the distance of a data point from the center of the sphere. Choosing a threshold for the attack scores determines the radius of the sphere enclosing normal data points.

4 Results

The supervised and the unsupervised algorithms are evaluated separately on the data with known and unknown attacks. The results are shown in Figs. 1 and 2 respectively. The ROC curves are averaged over 30 runs of each algorithm by fixing a set of false-positive rate values of interest and computing the means and the standard deviations of true-positive rate values over all runs for the values of interest.

The supervised algorithms in general exhibit excellent classification accuracy on the data with known attacks. The best results have been achieved by the C4.5 algorithm which attains the 95% true positive rate at 1% false-positive rate. The next two best algorithms are the MLP and the SVM, both non-linear, followed by the local k-Nearest Neighbor algorithm. The difference between the four best methods is marginal. The worst results were observed with the three linear algorithms. One can thus conclude that a decision boundary between the attack and the normal data in KDD Cup features is non-linear, and is best learned by non-linear algorithms or their approximations.

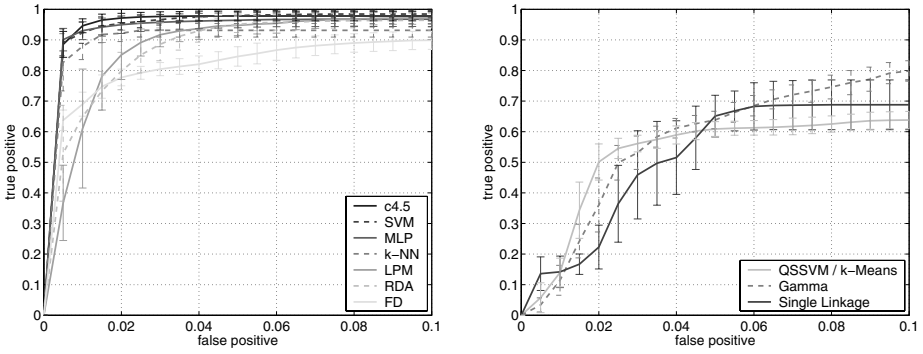

Fig. 1. ROC-curves obtained on *known* attacks: supervised (left) and unsupervised (right) methods

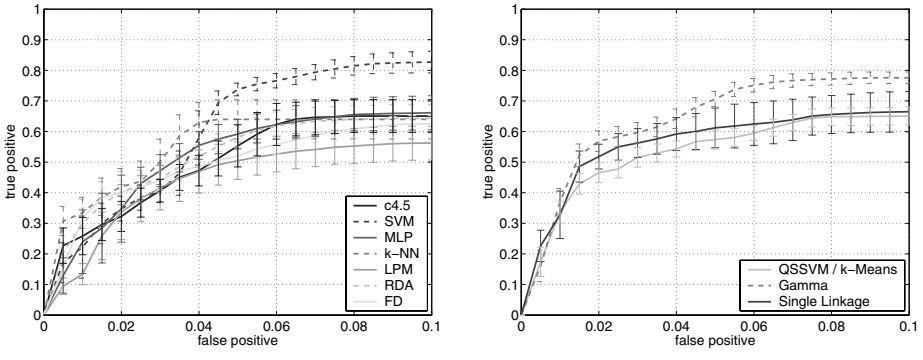

Fig. 2. ROC-curves obtained on *unknown* attacks: supervised (left) and unsupervised (right) methods

The accuracy of supervised algorithms deteriorates significantly if unknown attacks are present in the test data, as can be seen in the left part of Fig. 2. Not all algorithms generalize equally well to the data with unknown attacks. The best results (with a significant margin) are attained by the SVM, which can be attributed to the fact that the free parameters of this algorithm are motivated by learning-theoretic arguments aimed at maintaining an ability to generalize to unseen data. The next best contestant is the k-Nearest Neighbor algorithm which possesses the most similarity to the unsupervised methods. The remaining algorithm perform approximately equally.

The unsupervised algorithms exhibit no significant difference in performance between known and unknown attacks. This result is not unexpected: in fact, in all data sets the attacks are unknown to the algorithms – the two data sets differ merely in the set of attacks contained in them. Among the algorithms the preference should be given to the γ-algorithm which performs especially well on the "unknown" data set. The accuracy of unsupervised algorithms on both

data sets is approximately the same as that of supervised algorithms on the "unknown" data set.

5 Conclusions

We have presented an experimental framework in which supervised and unsupervised learning methods can be evaluated in an intrusion detection application. Our experiments demonstrate that the supervised learning methods significantly outperform the unsupervised ones if the test data contains no unknown attacks. Furthermore, among the supervised methods, the best performance is achieved by the non-linear methods, such as SVM, multi-layer perceptrons, and the rule-based methods. In the presence of unknown attacks in the test data, the performance of all supervised methods drops significantly, SVM being the most robust to the presence of unknown attacks.

The performance of unsupervised learning is not affected by unknown attacks and is similar to the performance of the supervised learning under this scenario. This makes the unsupervised methods, which do not require a laborious labelling process, a clear forerunner for practical purposes if unknown attacks can be expected.

Our findings suggest that the problem of test data being drawn from a different distribution cannot be solved within the purely supervised or unsupervised techniques. An emerging field of semi-supervised learning offers a promising direction of future research.

Acknowledgements. The authors gratefully acknowledge the funding from *Bundesministerium für Bildung und Forschung* under the project MIND (FKZ 01-SC40A), We would like to thank Stefan Harmeling and Klaus-Robert Müller for fruitful discussions and valuable help in the implementation of the algorithms.

References

1. Bace, R., Mell, P.: NIST special publication on intrusion detection systems. National Institute of Standards and Technology (2001)
2. Portnoy, L., Eskin, E., Stolfo, S.: Intrusion detection with unlabeled data using clustering. In: Proc. ACM CSS Workshop on Data Mining Applied to Security. (2001)
3. Eskin, E., Arnold, A., Prerau, M., Portnoy, L., Stolfo, S.: A geometric framework for unsupervised anomaly detection: detecting intrusions in unlabeled data. In: Applications of Data Mining in Computer Security. Kluwer (2002)
4. Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., Srivastava, J.: A comparative study of anomaly detection schemes in network intrusion detection,. In: Proc. SIAM Conf. Data Mining. (2003)
5. Laskov, P., Schäfer, C., Kotenko, I.: Intrusion detection in unlabeled data with quarter-sphere support vector machines. In: Proc. DIMVA. (2004) 71–82
6. Laskov, P., Schäfer, C., Kotenko, I., Müller, K.R.: Intrusion detection in unlabeled data with quarter-sphere support vector machines (extended version). Praxis der Informationsverarbeitung und Kommunikation **27** (2004) 228–236

7. Ghosh, A.K., Schwartzbard, A., Schatz, M.: Learning program behavior profiles for intrusion detection. In: Proc. of the 1st USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, USA (1999) 51–62 http://www.cigital.com/papers/download/usenix_id99.pdf.
8. Warrender, C., Forrest, S., Perlmutter, B.: Detecting intrusions using system calls: alternative data methods. In: Proc. IEEE Symposium on Security and Privacy. (1999) 133–145
9. Mukkamala, S., Janoski, G., Sung, A.: Intrusion detection using neural networks and support vector machines. In: Proceedings of IEEE Internation Joint Conference on Neural Networks. (2002) 1702–1707
10. Lee, W., Stolfo, S., Mok, K.: A data mining framework for building intrusion detection models. In: Proc. IEEE Symposium on Security and Privacy. (1999) 120–132
11. Stolfo, S.J., Wei, F., Lee, W., Prodromidis, A., Chan, P.K.: KDD Cup - knowledge discovery and data mining competition (1999). http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
12. Lippmann, R., Cunningham, R.K., Fried, D.J., Kendall, K.R., Webster, S.E., Zissman, M.A.: Results of the DARPA 1998 offline intrusion detection evaluation. In: Proc. RAID 1999. (1999). http://www.ll.mit.edu/IST/ideval/pubs/1999/RAID_1999a.pdf.
13. Lippmann, R., Haines, J.W., Fried, D.J., Korba, J., Das, K.: The 1999 DARPA off-line intrusion detection evaluation. Computer Networks **34** (2000) 579–595
14. Lee, W., Stolfo, S.: A framework for constructing features and models for intrusion detection systems. In: ACM Transactions on Information and System Security. Volume 3. (2001) 227–261
15. Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1992)
16. Duda, R., P.E.Hart, D.G.Stork: Pattern classification. second edn. John Wiley & Sons (2001)
17. Rojas, R.: Neural Networks: A Systematic Approach. Springer-Verlag, Berlin, Deutschland (1996)
18. Friedman, J.: Regularized discriminant analysis. Journal of the American Statistical Association **84** (1989) 165–175
19. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press, Cambridge, MA (2002)
20. Harmeling, S., Dornhege, G., Tax, D., Meinecke, F., Müller, K.R.: From outliers to prototypes: ordering data. Unpublished manuscript (http://ida.first.fhg.de/∼harmeli/ordering.pdf), submitted. (2004)